Learning in Robot Vision Directed Reaching: A Comparison of Methods

Michael R. Blackburn and Hoa G. Nguyen

Naval Command, Control and
Ocean Surveillance Center
Research, Development, Test, and
Evaluation Division
San Diego, CA 92152-7383

Abstract

Four neural network algorithms were examined for their ability to adaptively associate stereo camera coordinates with joint positions of a three degree of freedom manipulator arm in a 3D reaching task. Given reasonable numbers of training exemplars for an implementation in real hardware, all networks trained to significant errors. Two secondary error correction procedures were then tested. Both further reduced errors, but one method that depended on continuous visual and proprioceptive feedback to train a small set of associative weights that correlated joint and camera velocities was especially effective in eliminating errors. Stereo pan, tilt and vergence information was used to direct ballistic reaching, but relative depth information, was used for the visual feedback of end-effector velocity in the second error correction method.

1 Introduction

The problem addressed in this study is the one of directing the end-effector of a robotic manipulator arm onto a visually located target. This control problem is part of a larger issue in sensor-motor coordination. The problem is common to both natural systems (i.e. animals) and artificial systems (i.e. autonomous robots). It is now widely appreciated that this is a difficult problem because the systems are non-linear, the kinematics can be unknown, there can be excess degrees of freedom offering nonunique solutions, and the available state information can contain errors [Kawato, 1990; Oyama et al. 1991].

This problem is only important to the extent that it is desirable to have an autonomous robot that can see and manipulate objects in its environment. If we wish to replace man in some industries where the costs of labor exceed the benefits, then this and similar problems of sensor-motor coordination must be solved. According to Dickerson et al. [1990] material handling, including assembly and logistics contribute more to manufacturing costs than do any other activity. One way to reduce labor costs is through automation, yet the automation of material handling will depend on a solution to machine camera-manipulator coordination. One of the difficulties with camera control of a manipulator is establishing and maintaining the calibration of the two systems [Korde et al., 1992]. This is the first objective of the present work.

1.1 Non-adaptive Solutions

Because of the non-linearities the most common non-adaptive approach to camera-manipulator control is to partition the problem space in many smaller spaces and approximate solutions using linear control functions [Baker and Farrell, 1990]. While this piecewise linear approach minimizes the errors, it does involve a lot of human planning, processing, and fine tuning. Another inherent problem is that accuracy is limited to resolution; that is, to the size of the pieces into which the problem is broken. Most problematic is that the parameters of the functions are static when defined in advance of performance. Changes in system dynamics or kinematics will invalidate the parameters. Other non-adaptive methods of solution are variously available [Bennett et al., 1989; Hou and Utama, 1991] but still what is needed is the ability to meet unknown or unexpected changes in system dynamics or kinematics. We, therefore, turn to adaptive methods to define system kinematics and dynamics. These methods should demonstrate both stable learning of statistically significant relationships and short-term sensitivity to perturbations [Baker and Farrell, 1990].

1.2 Adaptive Solutions

Several researchers have already addressed the issue of adaptive camera-manipulator coordination [Coiton et al., 1991; Cooperstock and Milios, 1993; Hou and Utana, 1991; Kuperstein and Rubinstein, 1989; Mel, 1990; Li and Ogmen, 1994; Oyama et al., 1991; Ritter et al., 1988]. Generally, the objectives of these studies have

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **NOV 1994**	2. REPORT TYPE	3. DATES COVERED **00-00-1994 to 00-00-1994**
4. TITLE AND SUBTITLE **Learning in Robot Vision Directed Reaching: A Comparison of Methods**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Command, Control and Ocean Surveillance Center,Research, Development, Test, and Evaluation Division,San Diego,CA,92152-7383**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
Proc. 1994 ARPA Image Understanding Workshop, Monterey, CA, November 1994

14. ABSTRACT
Four neural network algorithms were examined for their ability to adaptively associate stereo camera coordinates with joint positions of a three degree of freedom manipulator arm in a 3D reaching task. Given reasonable numbers of training exemplars for an implementation in real hardware, all networks trained to significant errors. Two secondary error correction procedures were then tested. Both further reduced errors, but one method that depended on continuous visual and proprioceptive feedback to train a small set of associative weights that correlated joint and camera velocities was especially effective in eliminating errors. Stereo pan, tilt and vergence information was used to direct ballistic reaching, but relative depth information, was used for the visual feedback of end-effector velocity in the second error correction method.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**8**	

been to define the inverse kinematics of the system through experience with system performance, incorporating the effects of unknown factors that frustrate modeling and prediction. The three most common algorithms used in the learning controllers have been Back Propagation (BP), the Kohonen Self-Organizing Map (SOM), and a third method, encompassing a variety of methods, that involved the partition, usually explicitly, of the work space, and the assignment of different parameters to govern the mapping when operating within each partition. This last category of methodology is similar in concept to the piecewise linear approach taken by non-adaptive solutions. The adaptive advantage of the third approach is based on the way in which the parameters of each sector are established.

The circular reaction learning protocol was widely used in these studies. This protocol allows a form of unsupervised learning that none-the-less takes advantage of the presence of a desired output as a reference for learning. In circular reaction learning, the manipulator arm randomly assumes configurations that bring the end effector into view. For each configuration of the manipulator, the vision system locates the end effector, and correlations between joint variables and camera variables are learned. After the correlations are coded in connection weights, the input of camera position information resulting from a visually located target can evoke joint positions that will bring the end effector onto the target, approximating solutions to the inverse kinematics.

Several groups, dissatisfied with the degree of accuracy obtainable from the coarse coded representations of visual space and joint space, implemented additional error correction methods to fine-tune the visual control of end-effector location [Cooperstock and Milios, 1993; Kuperstein and Rubinstein, 1988; Ritter et al., 1989; Li and Ogmen, 1994;]. Some of these methods were themselves adaptive. Kuperstein and Rubinstein [1988] attempted to correct the original weights that associate the joint positions with the visual coordinates of the target. Ritter et al [1989] trained additional weights to represent the first derivative of the function relating joint angles to the visual coordinates. Closed loop visual feedback is necessary for these methods of correction.

2 Current Work

The present work evaluates four neural network learning algorithms and two methods of secondary error correction on a model of visual-motor coordination in three dimensions. Functions that relate activity on log-polar transformations of the stereo visual input to camera and arm motor commands are introduced.

2.1 The Model

A drawing of the configuration of manipulator arm and stereo cameras that was used in the simulations to test the algorithms is shown in Figure 1. The cameras are mounted on a stereo pan, tilt, and vergence mechanism that provides data on these three camera orientation parameters. The three active joints of the manipulator arm: shoulder rotate, shoulder elevate, and elbow bend, determine end effector location, which is acquired by the vision system. The joints are restricted to motions through 180 degrees.

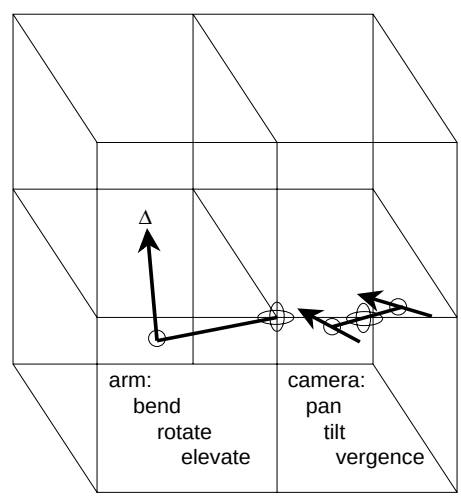

Figure 1. Configuration of manipulator arm and stereo cameras in simulation.

2.2 An Explicit Look-up Table Method

Probably the simplest way to produce joint angles from camera angles is to store sets of angle data recorded from the joints and pan/tilt/vergence mechanism when the camera is focussed on the end-effector. Then, given any camera focus point, the stored camera angles that match most closely the current values can be found, and the associated

joint angles may be recovered. Of course, the more sets of angles that are stored, the more likely it will be that any given camera focus point will have a stored joint set close to its actual joint set.

To assess the accuracy of the look-up table method using the arm-camera model of Figure 1, N pairs of correlated joint angles and camera angles were randomly generated and stored. Then M pairs of new joint angles and camera angles were randomly generated and tested against the stored pairs. The error was measured by the end effector location disparity between the stored location and the location resulting from the new configuration. Results for N = 4000, 1000, 250, 100, and 62 and M = 1000 are shown in Figure 3. If one has the patience to record 4000 positions of the arm and camera, an average error of 3.3 cm in end effector location can be expected for a manipulator arm similar to the model of Figure 1 when matching new camera orientations with the stored values and moving the arm to the correlated and stored joint positions. Practically, many fewer positions can be acquired using hardware in a reasonable amount of time. The results of Figure 3 establish a standard that must be bettered by any algorithm that attempts to calculate joint angles given camera orientation for ballistic reaching to a visually identified target.

2.3 Neural Network Methods to Direct Ballistic Reaching

We studied four neural network methods to see if joint angles could be produced given camera orientation with greater accuracy for a given size of training data than that which could be achieved with a simple look-up table. The four methods were: 1) a three layer perceptron with back propagation learning (BP), 2) a Kohonen self-organizing map to generalize correlations of joint angles and camera angles (SOM), 3) a two layer perceptron with delta rule learning and preprocessing using vertex normal features extracted from the joint and camera parameters (PVN), and 4) a two layer perceptron with delta rule learning and population coding using a geometric distribution of the joint and camera parameters (PPC).

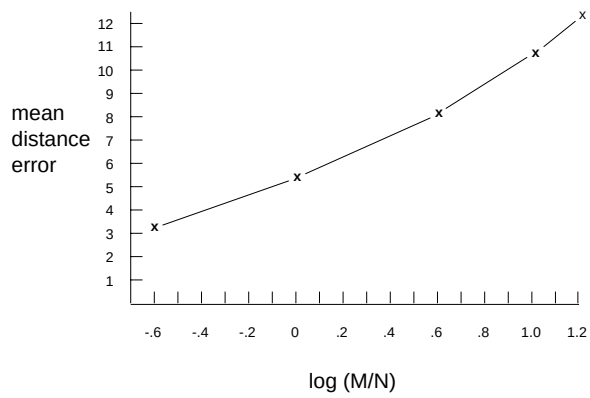

Figure 3. Averages of M distance errors of end effector from target using look-up tables of N stored vectors.

2.3.1 Back Propagation

We used a simple form of back propagation (Figure 4) taken from Rumelhart et al. (1986). The input layer contained 3 elements, one for each degree of freedom of the stereo cameras. There was one hidden layer of 7 elements. The output layer contained 3 elements, one for each joint. The input and training data were normalized to their range. A threshold element provided input to both the hidden and output layers through modifiable connections that were trained as the other elements. The constant threshold bias was set at 0.5. The learning rate was set at 0.05. A momentum factor was not used. Weights were initialized to random numbers between -0.05 and +0.05.

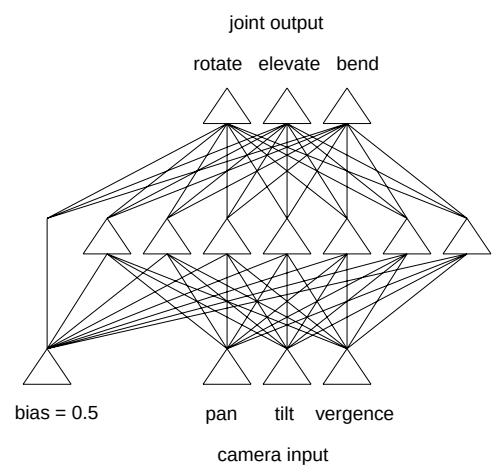

Figure 4. Three layer perceptron with back propagation learning. Connections between the input and hidden layer and between the hidden and output layer are modifiable.

2.3.2 Kohonen Self-Organizing Map

The Self-organizing map algorithm that we used (Figure 5) closely paralleled that described by Kohonen [1990]. The input vector was composed of the three joint angles and the three camera position angles for pan, tilt, and vergence. All data were normalized to their range. After creation of the map, joint angles were recalled by finding the best match for the three input vector elements belonging to the camera angles, and then reading off the three matched weights belonging to the joints. The dimensions of the weight matrix were 16*16*6. Weights were initialized to random numbers between -0.01 and +0.01.

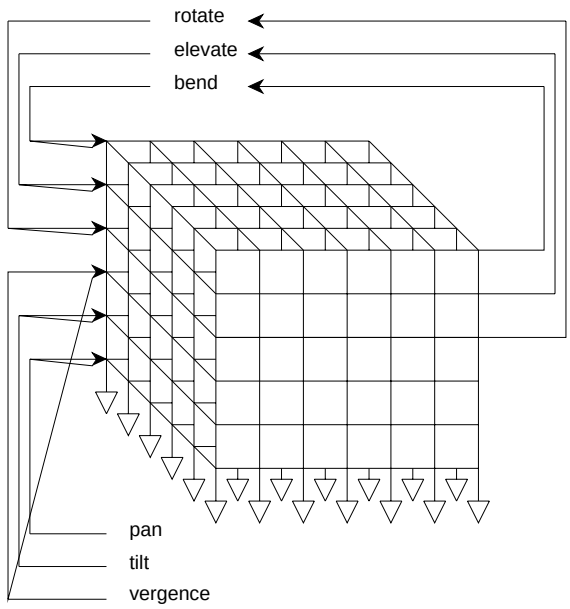

Figure 5. Kohonen self-organizing map. All processing elements receive a copy of the input. After completion of the learning phase, the element, whose camera portion of the input weight vector best matches the camera input, is selected. The joint portion of its input vector is taken as the output. The dimensions of this figure are reduced from those used in the simulation for clarity of presentation.

2.3.3 Vertex Normal Features

The two layer perceptron with delta rule learning and preprocessing using vertex normal features was developed in this lab to remove the need for back propagation of errors during learning. Recognizing that the weights between the input layer and the hidden layer in a three layer perceptron with BP learning essentially create feature filters of the input space, the use of a preprocessor that provides this feature extraction process would eliminate the need for one layer of modifiable weights. This should greatly increase learning rates in the remaining pathway between hidden layer and output.

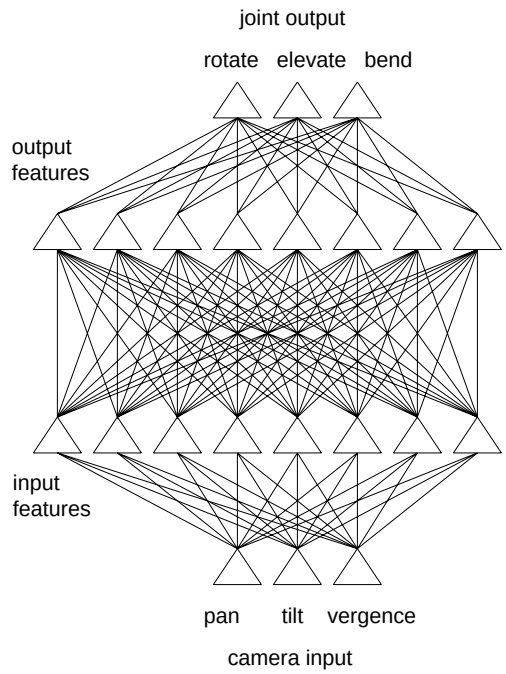

Figure 6. Two layer perceptron with vertex normal feature preprocessing. Only connections between the input and output feature layers are modifiable.

The preprocessor was a vertex normal representation of the input [Williams, 1986]. The vertex normal representation is similar to the functional link architecture [Klassen et al.,1988], and to the higher order unit of Fahner [1990], but completely characterizes the input space. In the present application, the three input variables for camera orientation were transformed to eight features by first normalizing each to unit length, then taking all joint products among the variables and their complements. The eight features for camera orientation were $(1.0-p)*(1.0-t)*(1.0-v)$, $p*(1.0-t)*(1.0-v)$, $t*(1.0-p)*(1.0-v)$, $p*t*(1.0-v)$, $v*(1.0-p)*(1.0-t)$, $v*p*(1.0-t)$, $v*t*(1.0-p)$, $p*t*v$.

Joint angles were also submitted to the vertex normal preprocessor. Mapping was accomplished by correlating, using the Delta Rule, these two sets of features through 64 modifiable connections. For the output, the predicted set of features were

recombined by a process that is the inverse to feature creation. The learning rate was set to 1.0. Weights were initialized to 0.0. Figure 6 shows the network for this process.

2.3.4 Population Coding

We developed a second method of feature definition that partitioned the scalar input values into vectors of processing elements. One method of partitioning is to explicitly represent the 3D work-space with a 3D matrix of elements [Coiton et al., 1991; Li and Ogmen,1994]. This, however, results in a large matrix that can require substantial processing. Instead we represented each joint and camera angle by a small number of processing elements, each element maximally sensitive to particular angle, with sensitivity falling off as the angle differs from this preferred value. Elements in the vector share activity in proportion to the proximity of the value to their positions in the vector. Figure 7 demonstrates this process.

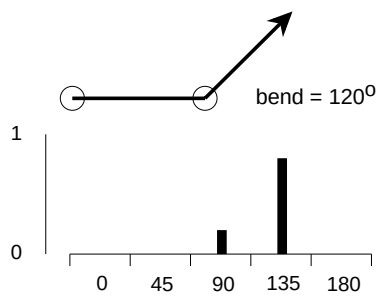

Figure 7. Population coding of joint position on a five-element pool. Each element has a center angle to which it is maximally sensitive.

In the present application, after normalization to its range, each variable was partitioned into a vector of 5 elements. A three dimensional matrix, containing 5*5*5 elements, was constructed for each set of input and output vectors. The activity on any element in the matrix is the product of the activities of the vector elements that intersect upon it. Each input angle vector will have almost always two active elements, thus the input matrix will have 8 active elements, resulting from the three dimensional intersection of the contributing vectors.

Matrices were formed for both joint and camera variables. A weight matrix associated the two. Weights were initialized to 0.0. Connection weights were trained using the common Delta Rule with learning rate set to 1.0. After a period of training, the output matrix could contain many active elements with any given input vector. Reconstruction of each output vector was accomplished by summation over all other dimensions of the matrix for each element of the output vector. Reconstruction of the individual variables was accomplished by taking the vector sum of the elements in each output vector. A form of population coding is achieved by this method.

2.4 Methods for Secondary Error Correction Under Continuous Visual Feedback

Two methods were explored for fine sensor-motor control under visual feedback. The first is related to the procedures of Kuperstein and Rubinstein (1988) which we call Error Compensation (EC), while the second is related to procedures of Ritter et al. (1989) which we call Visual Servo Control (VSC).

2.4.1 Error Compensation

For the method of Error Compensation, after the conclusion of the ballistic reach, the vision system executed a saccade to the end effector and calculated the motor command that would have occurred if the target and not the end-effector was at that location. Generally the difference in the original and second motor commands can be used to correct the original motor command and improve target reaching. The correction, if successful, may be saved by modifying the weights by the error in motor commands given the end effector coordinates.

2.4.2 Visual Servo Control

In the method for Visual Servo Control a second set of nine weights were trained to associate changes in joint angle with observed changes in end-effector position relative to target location. Each weight associated the expected change in joint position with change in end-effector location on either the *x*, *y* or *z* axes. The weights were modified by randomly perturbing the joint positions after conclusion of the ballistic reaching and noting the change in end effector location with the change in joint position. If the end-effector was

closer to the target than before the perturbation, the weights were increased, if not, the weights were decreased. The joint position was corrected by the product of its end-effector location and the associated weight.

3 Simulation Results

3.1 Ballistic Reaching

All algorithms were trained and tested under the same conditions, except for learning rates, which were optimized for each algorithm. The method of circular reaction was used for training. One hundred correlated pairs of joint and camera parameters were generated and presented to the networks for 1000 repetitions during learning. Afterward, 100 new correlated pairs of joint and camera parameters were generated. The camera orientation, measured as pan, tilt, and vergence, was acquired by allowing the vision system to saccade to the target [Blackburn, 1993]. The camera orientation information was passed through the network weights to produce joint configurations. The end effector locations of these new configurations were then compared with the locations correlated with the input camera orientations, and an error was calculated as in the look-up table method.

Table I shows the relative performance of the four neural network algorithms along with the look-up table method. The average distance errors in centimeters for these 100 test trials are given. Simulation time is provided for a relative comparison of the computer time required to train the networks.

The relative errors and training times listed in Table I are relevant only to the present task and training conditions. Back propagation trains slowly and was disadvantaged by the 1000 cycle limit during training. Given 300,000 cycles of the same input data, back propagation reduced the average error to 5.13 cm. The 1000 repetitions of the training data actually impaired the performance of the PPC network compared to its average error after 100 training repetitions of only 5.54 cm.

Table I

Average Errors in Ballistic Reaching

Algorithm	error	time	# weights
Look-up	10.8	0:06	N/A
BP	13.4	1:05	52
SOM	8.9	14:19	1536
PVN	5.7	1:11	64
PPC	7.8	2:30	15625

3.2 Secondary Error Correction

The data in Table II were acquired under somewhat different conditions from the data in Table I. Because in a hardware implementation the robot would be required to be operational for all of the learning involving visual feedback of end-effector position, the number of trials used in our simulation training was reduced to 1000 single presentations of random exemplars. After training, the algorithms were tested with 100 additional random trials without additional learning.

Table II

Errors in Reaching after Secondary Correction

Algorithm	before error	after error	# wts
PVN + EC	9.72	8.55	0
PPC + EC	4.37	2.82	0
PVN + VSC	9.92	0.15	9
PPC + VSC	5.00	0.33	9

Using the EC algorithm, modifications to the weights that influence ballistic reaching are not always successful, and seem to depend upon the network that learned the ballistic reaching task. Improvements to the PPC performance of approximately 40% are possible, yet the weight modifications do not work at all with the PVN network. When using the PVN model, the EC algorithm can reduce ballistic reaching errors if the weights are not modified by the process. The improvement in PVN performance with unsaved EC modifications to joint position was 23%.

The VSC algorithm worked well with both the PPC and PVN networks, reducing errors to less than 0.5 cm on the average after training on the 1000 reaching trials.

4 Discussion

In depth perception for eye-hand coordination at close range (i.e. at arm's length) both absolute depth information for the target and relative depth information for the hand in the vicinity of the target are important. The target is acquired and maintained on the center of the receptor surface by saccadic and pursuit eye movements. The absolute depth of the target, estimated by the degree of vergence, initiates and sets the amplitude of the ballistic reach. As the end-effector approaches the target, it enters the visual field. Then the position of the end-effector relative to the target tells the system how to correct the error to capture the target.

We must consider the application of the adaptive control algorithms to real hardware. One of the first problems encountered using hardware that is not obvious with simulations is that the visual identification of the target is noisy. A second problem is that the hardware, manipulator arm as well as cameras moving on a pan/tilt/vergence mechanism, require a finite amount of time to execute movements. Thus a large number of trials with moving equipment will consume a lot of time. Fortunately, the algorithms do not require a physical repetition of each configuration on each cycle. Once a representative number of correlated joint configurations and camera orientations have been performed and saved in a data file, the networks can be trained off-line, with the hardware asleep. The PPC algorithm requires the least amount of off-line training while BP requires the most.

Both BP and SOM are sensitive to initial conditions, that is, the connections weights must be set to small random values. This initial randomization biases the sensitivities of the network elements to the input pattern. Some initial conditions result in better performance than do others. The optimal initial conditions must be determined empirically at present. PVN and PPC are intialized with zero weights, yet these networks can be trained from any set of initial weights. Learning in those two-layer perceptron networks depends only upon the performance error. This is a distinct advantage when retraining is necessitated by some change in the kinematics or calibration of the system.

The EC method of secondary error correction can proceed in parallel with the ballistic error correction procedures as long as the end effector is accessible to the vision system. Learning is slowed by this procedure, however, due to the requirements that the manipulator arm must actually move and the cameras saccade to the end location on each learning trial.

The VSC method also requires that the arm move and be observed, thus potentially slowing the learning process. However, the VSC method of secondary error correction can progress quite independently of the ballistic reaching network and its training. Learning can progress in two stages. First, after collecting the sample of representative target locations and arm configurations, ballistic learning can continue off-line. Second, after the ballistic learning stage, the VSC learning can be accomplished on-line with a relatively small number of trials.

The question of the optimal algorithm and training conditions is obvious. If memory is limited the look-up-table methods, including the PPC model, are inappropriate. If on-line adaptation is required, SOM and BP are inappropriate. Repetition of a set of exemplars improves BP and PVN performance, but this is not practical if the EC algorithm is used additionally. The best ballistic reaching is achieved at the lowest cost using the PVN algorithm without any additional error correction procedure, allowing many silent repetitions of the input data. The best additional error correction procedure is the VSC algorithm which works well independently of the ballistic reaching as long as the end-effector is located in the vicinity of the target and its relative velocity can be determined.

It is instructive that the surest method is to fill a large array of correlated pairs of joint and camera angles, then look up the best match, or best eight matches, during recall, and interpolate or average the results. It may be that all of the neural networks achieved success by essentially this procedure. Surely, those that partition the input space such as SOM and PPC do so. But even accuracy in back propagation learning is known to depend upon the appropriate number of hidden layer elements. No one has yet been able to specify what that number should be for all applications, but it may relate to the degree of non-monotonicity of the input variables, that is, its nonlinearity. The hidden layer, by representing features, partitions the input space. Thus, the hidden layer selects the

output weights that belong to the particular function that describes the relationships of the variables in their current range. The Kohonen SOM does this as well. We could argue that this is the role of feature detectors in general

Acknowledgement

This research is supported by the Advanced Research Projects Agency and the Office of Naval Research under contract number N0001493WX2D002.

References

Baker, W.L. and Farrell, J.A. [1990] Connectionist learning systems for control. SPIE Proceedings 1382, 181-198.

Bennett, D.J., Hollerbach, J.H., and Geiger, D. [1989] Autonomous robot calibration for hand-eye coordination. *Robotics Research*, Cambridge, MA: MIT Press, 137-144.

Blackburn, M.R. [1993] Machine visual targeting modeled on biological reflexes. NRaD TD 2455, Naval Command, Control and Ocean Surveillance Center RDT&E Division, San Diego, California.

Coiton, Y., Gilhodes, J.C., Velay, J.L. and Roll, J.P. [1991] A neural network model for the intersensory coordination involved in goal-directed movements. *Biological Cybernetics, 66*, 167-176.

Cooperstock, J.R. and Milios, E.E. [1993] Self-supervised learning for docking and target reaching. *Robotics and Autonomous Systems, 11*, 243-260.

Dickerson, S.L., Lee, K.-M., Lee, E.H., Single, T. and Li, D. [1990] Intelligent material handling - use of vision. SPIE Proceedings 1381, 201-207.

Fahner, G. [1990] A higher order unit that performs arbitrary Boolean functions. IJCNN 90 Proceedings, III, 193-197.

Hou, E.S.H. and Utama, W. [1991] An artificial neural network for redundant manipulator inverse kinematics computation. SPIE Proceedings 1607, 668-677.

Kawato, M. [1990] Computational schemes and neural network models for formation and control of multijoint arm trajectory. In W.T. Miller, III, R.S. Sutton and P.J. Werbos (eds.) *Neural Networks for Control*, Cambridge, MA: MIT Press, 197-228.

Klassen, M., Pao, Y.H. and Chen, V. [1988] Characteristics of the functional link net: A higher order delta rule net. IEEE ICNN 88 Proceedings, I, 507-513.

Kohonen, T. [1990] The self-organizing map. Proceedings of the IEEE, 78, 1464-1480.

Korde, U.A., Gonzales-Galvan, E. and Skaar, S.B. [1992] Three-dimensional camera-space manipulation using servoable cameras. SPIE Proceedings 1825, 658-667.

Kuperstein, M. and Rubinstein, J. [1989] Implementation of an adaptive neural controller for sensory-motor coordination. IJCNN 89 Proceedings, II, 305-310.

Li, L. and Ogmen. H. [1994] Visually guided motor control: Adaptive sensorimotor mapping with on-line visual-error correction. Proceedings of the World Congress on Neural Networks, June 5-9, 1994, San Diego, CA, II, 1127-134.

Mel, B.W. [1990] *Connectionist Robot Motion Planning.* Boston, MA: Academic Press.

Oyama, E., Maeda, T. and Tachi, S. [1991] A study of human hand position control learning output feedback inverse model. IJCNN 91 Proceedings, II, 1434-1443.

Ritter, H.J., Martinetz, T.M. and Schulten, K.J. [1988] *Neural Networks, 2*, 159-168.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J. [1986] Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland (eds.), *Parallel Distributed Processing, I*, Cambridge, MA: MIT press, 319-362.

Williams, R.J. [1986] The logic of activation functions. In D. E. Rumelhart and J.L. McClelland (eds.) *Parallel Distributed Processing, I*, Cambridge, MA: MIT Press, 423-443.